Docket No.: 042390.P8815

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Stephen S. Chang Application No.: 09/608,869 Filed: June 30, 2000 For: TASK-BASED MULTIPROCESSING SYSTEM	Examiner: Majid A. Banankhah Art Group: 2127

APPEAL BRIEF

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

Applicant submits the following Appeal Brief pursuant to 37 C.F.R. § 1.192 for consideration by the Board of Patent Appeals and Interferences. Applicant also submits herewith our check number 31448 in the amount of $340.00 to cover the cost of filing the opening brief as required by 37 C.F.R. § 1.17(f). Please charge any additional fees or credit any overpayment to our deposit Account No. 02-2666. A duplicate copy of the Fee Transmittal is enclosed for this purpose.

11/02/2004 RMEBRAHT 00000030 09608869
02 FC:1402 340.00 OP

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST

The real party in interest is the assignee, Intel Corporation.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences known to the appellants, the appellants' legal representative, or assignee, which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

III. STATUS OF CLAIMS

Claims 1-30 of the present application are pending and remain rejected. The Applicant hereby appeals the rejection of claims 1-30.

IV. STATUS OF AMENDMENTS

The Applicant filed an amendment on June 21, 2004, in response to a Final Office Action issued by the Examiner on April 20, 2004. In response to the June 21, 2004 amendment, the Examiner issued an Advisory Action on August 4, 2004. The Applicant filed a Notice of Appeal from the Advisory Action issued by the Examiner on August 20, 2004.

V. SUMMARY OF CLAIMED SUBJECT MATTER

1. Independent claims 1, 11, and 21:

A task manager used internally or externally to a processor to manage tasks executed by the processor or processors[1]. The task manager includes a task table, a block allocation circuit, and a task coordinator[2]. The task table stores task entries corresponding to tasks executed by at least one of the processors. The task table provides a means to keep track of the tasks obtained or executed by the processors[3].

The block allocation circuit allocates blocks of the cache memory used by the tasks. The block allocation circuit is interfaced to the bus interface and the internal cache

[1] Specification, page 4, lines 13-18; page 4, line 27; page 5, lines 1-2, lines 6-9, lines 17-21; Figure 1, reference numbers 114_i.., 114_N, 127_i..,127_N, 155

[2] Specification, page 7, lines 9-11; Figure 2A, references 210, 220, and 230

memory or the external cache memory. The block allocation circuit monitors the usage of data blocks in the cache memory and locates available blocks when requested by a task. The block allocation circuit searches for a free block of cache that satisfies the task requirement[4]. The task coordinator coordinates the tasks in response to a task cycle issued by at least one of the processors. The task coordinator interfaces to the processor core and the internal cache memory or the external cache memory to coordinate activities of the tasks[5].

2. Dependent claims 2, 12, and 22:

The task table includes L task entries organized into a status field, a task ID field, a start address field, a task block size field, and a cache address field[6]. Each of the task entries contains information about a task. The status field indicates the status of a task. The status may be modified, invalid, shared, and exclusive. The task ID field contains the task ID of the task. This task ID is unique to the task in the active session in which tasks are running. The start address field contains the start address of the data block associated with the task. The task block size field contains the task block size. The cache address field contains the start physical address of the cache memory that stores the data block associated with the task[7].

3. Dependent claims 3, 13, and 23:

The task cycle format includes a task identifier (ID), a task command, a start address, and a task block size[8]. The task ID is used to identify or name a task in a unique manner within a processing session. The task command specifies the type of operation to be operated on a task. The start address specifies the starting address of the data block associated with the task. The task block size specifies the size of the data block in terms of some size unit such as the cache line[9].

4. Dependent claims 4, 14, and 24:

The block allocation circuit includes a search logic circuit and a block information generator[10]. The search logic circuit determines a list of candidates free blocks of memory by scanning through the busy flag signals provided by the busy flag generator. The busy

---

[3] Specification, page 7, lines 13-16
[4] Specification, page 7, lines 19-25; page 8, lines 1-2
[5] Specification, page 8, lines 3-7
[6] Specification, page 11, lines 9-11; Figure 4, reference numbers 420, 430, 440, 450, and 460
[7] Specification, page 11, lines 12-24
[8] Specification, page 10, lines 11-13; Figure 3, reference numbers 310, 320, 330, and 340
[9] Specification, page 10, lines 14-27; page 11, lines 1-17

flag indicates if the resource (e.g., memory) is busy or free. The block information generator provides the block information such as the block size, the starting address and the ending address of the block[11].

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Claims 1, 11, and 21 stand rejected under 35 U.S.C. §102(a) as being anticipated by U.S. Patent No. 6,128,307 issued to Brown ("Brown").
2. Claims 2-3, 12-13, and 22-23 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Brown in view of U.S. Patent No. 6,173,385 issued to Tuma et al. ("Tuma").
3. Claims 4-10, 14-20, and 24-30 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Brown in view of Tuma and further in view of U.S. Patent No. 5,138,696 issued to Nagata ("Nagata").

VII. ARGUMENTS

A. Claims 1, 11, and 21 Are Not Anticipated by Brown.

The Examiner rejected claims 1, 11, and 21 under 35 U.S.C. §102(a) as being anticipated by U.S. Patent No. 6,128,307 issued to Brown ("Brown"). Applicant respectfully traverses the rejection and contends that the Examiner has not met the burden of establishing a prima facie case of anticipation. To anticipate a claim, the reference must teach every element of a the claim. "A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." Vergegaal Bros. v. Union Oil Co. of California, 814 F.2d 628, 631, 2 USPQ 2d 1051, 1053 (Fed. Cir. 1987). "The identical invention must be shown in as complete detail as is contained in the...claim." Richardson v. Suzuki Motor Co., 868 F.2d 1226, 1236, 9 USPQ 2d 1913, 1920 (Fed. Cir. 1989).

Brown discloses a programmable data flow processor for performing data transfers. A system includes a data flow processor (DFP), a micro-controller unit (MCU) coupled to

---

[10] Specification, page 12, lines 22-23; Figure 5, reference numbers 510 and 530
[11] Specification, page 12, lines 24-26; page 13, lines 1-22

the DFP, and a digital signal processor (DSP) coupled to the DFP (Brown, col. 2, lines 55-59). The DFP selectively allows access to the memory by the MCU (Brown, col. 2, lines 63-64). The MCU is operable to transfer a task list to the memory which comprises tasks to be executed by the DSP (Brown, col. 3, lines 5-7). The MCU programs the DFP for a data stream to the DSP including the memory management (e.g., allocation of data buffers and streams for each task) (Brown, col. 9, lines 60-67; col. 10, lines 7-8). The DSP executive routine performs management of the task list using the data streams (Brown, col. 14, lines 7-8).

Brown does not disclose, either expressly or inherently, (1) a block allocation circuit to allocate blocks of cache memory used by at least one of the task, and (2) a task coordinator to coordinate the tasks in response to a task cycle issued by the processor.

Brown merely discloses an executive routine (EXEC) for memory management, not a block allocation circuit. The EXEC routine is the operating system that interfaces to and manages the individual signal processing block of code (Brown, col. 11, lines 59-61). It is generally programmed by the MCU (Brown, col. 11, line 67). Since it is merely a program, it is not a circuit coupled to the bus interface and a cache memory. Furthermore, the EXEC routines merely allocates the data buffers and data streams. It does not allocate blocks of cache memory. The Examiner further states that Brown disclose a task coordinator to coordinate the tasks in response to a task cycle issued to the processor. Applicant respectfully disagrees. Brown merely discloses a timer to provide a computer-triggered output to other blocks by a direct signal or a write/read (W/R) cycle through the DFP (Brown, col. 11, lines 1-3). A W/R cycle is not a task cycle issued by a processor. As described in the Specification, a task cycle transfers the information and data of a task on the system bus (See Specification, page 9, line 13). It may be either of loading a task or of executing a task (See Specification, page 14, lines 2-3; lines 8-25).

In the final Office Action, the Examiner made several counter-arguments to Applicant's arguments. Applicant addresses these counter-arguments in the following.

(i) Block allocation circuit and task coordinator:

In the final Office Action, the Examiner cited Brown at col. 10, lines 48-57 to support the contention that Brown discloses a block allocation circuit, and Brown at col. 6, lines 62-65 to support the contention that Brown discloses a task coordinator (Final Office

Action, page 2, paragraph 4(i)). Applicant respectfully disagrees. First, Applicant notes that by citing Brown at column 10, lines 48-57, and column 6, lines 62-65, the Examiner has changed position from the previous Office Action when the Examiner cited Brown at column 11, lines 59-61 regarding the EXEC routine and column 11, lines 1-3 regarding a timer.

Regarding the block allocation, Brown merely discloses the digital signal processor (DSP) determining a buffer allocation in the memory (Brown, col. 10, lines 52-53). The buffer is used to store the data streams. The DSP communicates to the data flow processor (DFP) that the buffer is available or full (Brown, col. 10, lines 53-57). A buffer allocation is not a block allocation. Furthermore, Brown does not disclose a block allocation circuit. A block allocation circuit allocates blocks of cache memory used by the tasks as recited in claims 1, 11, and 21.

Regarding the task coordinator, Brown merely discloses an interrupt routine handling the interface to the DSP from the DFP and passes control data along to the executive routine (Brown, col. 6, lines 63-65). In contrast, a task coordinator coordinates the tasks in response to a task cycle by the at least one processor as recited in claims 1, 11, and 21. An interrupt routine is invoked only when there is an interrupt and when the interrupt is enabled. The interrupt routine can only respond to the interrupting device that causes the interrupt. It does not coordinate the tasks in response to a task cycle.

(ii) Product and process claim language:

The Examiner states that if the product is the same as or obvious from a product of the prior art, the claim is unpatentable even the prior product was made by a different process (Final Office Action, page 3, paragraph 4(ii)). However, Applicant did not simply argue that Brown discloses a program. The main contention was that the EXEC routine does not perform a block allocation function. It does not allocate blocks of cache memory.

(iii) Interrupt routine and Executive routine:

The Examiner cited Brown in column 9, lines 65-68 to column 10, lines 1-8, to support the contention that Brown discloses a block allocation circuit and a task coordinator (Final Office Action, page 3, paragraph 4(iii)). As mentioned earlier, this

seems to be inconsistent with the Examiner's contention presented in (i) above. However, assuming the Examiner takes dual positions, this second characterization also fails to support an anticipation rejection. First, storing information on each task is not the same as allocating blocks of cache memory and/or coordinating tasks in response to a task cycle. The Examiner failed to identify the teaching of a task cycle and coordination of tasks in response to the task cycle. Second, the Examiner failed to identify the allocation of blocks of cache memory taught by Brown.

In the Advisory Action dated August 4, 2004, the Examiner stated that a block is a section of RAM temporarily allocated to a program or a task. However, this is not the task that is characterized by a task table having task entries. Furthermore, Brown does not disclose a block allocation circuit to allocate blocks of the cache memory. The Examiner further stated that diagram of a RAM interface is shown in Figurer 10, apparently referring to Figure 10 of Brown. However, this RAM interface does not show block allocation circuit, or a task coordinator. The RAM interface in Brown does not show cache memory used by at least one of the tasks.

(iv) Claim interpretation: (Final Office Action, page 3, paragraph 4(iv))

Claims should be interpreted consistently with the specification, which provides content for the proper construction of the claims because it explains the nature of the patentee's invention. See Renishaw P.L.C v. Marposs Societa Per Azioni, 158 F.3d 1243 (Fed. Circ. 1998). During patent examination, the pending claims must be "given the broadest reasonable interpretation consistent with the specification". See MPEP 2111. The terms "task", "task entries", "task table", "block allocation circuit," "task coordinator" and "task cycle" should be interpreted consistently with the specification such as provided on page 3 (lines 15-20, lines 23-26), page 7 (lines 12-25), page 8 (lines 1-7), page 9 (lines 3-13), and page 14 (lines 2-3, lines 8-23).

The Examiner failed to interpret the claims consistently with the specification. First, the word "task" used by Brown merely indicates an algorithm (Brown, col. 10, lines 1), not a partitioned program routine viewed as an instruction block with or without a data block, and that can be executed in parallel with other tasks (see specification, for example, page 3, lines 14-25). Second, the task status is not the same as a task table. As shown in Figure 24 of Brown, task status merely indicates whether a task is ready. In contrast, a

task table contains the states of active tasks (See, for example, Specification, page 3, line 17). Third, the issues of task coordinator and task cycle have already been discussed earlier.

Therefore, Applicant believes that independent claims 1, 11, 21 and their respective dependent claims are distinguishable over the cited prior art references.

B. Claims 2-3, 12-13, and 22-23 Are Not Obvious over Brown in View of Tuma.

The Examiner rejected claims 2-3, 12-13, and 22-23 under 35 U.S.C. §103(a) as being unpatentable over Brown in view of U.S. Patent No. 6,173,385 issued to Tuma et al. ("Tuma"). Applicant respectfully traverses the rejection and contends that the Examiner has not met the burden of establishing a prima facie case of obviousness. To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. *MPEP §2143, p. 2100-129 (8th Ed., Rev. 2, May 2004).* Applicants respectfully contend that there is no suggestion or motivation to combine their teachings, and thus no *prima facie* case of obviousness has been established.

Brown discloses a programmable data flow processor for performing data transfers as discussed above.

Tuma discloses an address generator for solid state disk drive. An address generation circuit includes cascaded shift registers to provide the basis for a multiplier circuit (Tuma, col. 3, lines 34-37). The multiplication process propagates a ripple carry to a last flip-flop and loads the multiplied address as the start address into a counter (Tuma, col. 4, lines 9-17).

Brown and Tuma, taken alone or in any combination, does not disclose, suggest, or render obvious (1) a block allocation circuit to allocate blocks of cache memory used by at least one of the task, (2) a task coordinator to coordinate the tasks in response to a task cycle issued by the processor; (3) task entry comprising at least one of a task status, a task identifier, a task start address, a task block size, and a task cache address, and (4) task

cycle providing at least the task IP, a task command, the task start address, and the task block size.

Brown does not disclose or suggest a block allocation circuit and a task coordinator as discussed above. Tuma merely discloses an address generator for a SCSI peripheral device. The stating address is used for reading or writing a logical block (Tuma, col. 4, lines 17-21). This is not the same as the task start address corresponding to a task.

The Examiner states that finding the start and end address of a block of memory for read and write is not different from finding the starting and ending address of a task (Final Office Action, page 4, paragraph 4(vi)). Applicant respectfully disagrees. First, the Examiner takes the phrase out of the context. Within the context of the block information of a free block available for a new task, Tuma does not disclose or suggest the block information. Second, Tuma merely discloses generating address for reading and writing a logical block, not starting and ending addresses associated with a task. The logical block size is selected by the host computer (Tuma, col. 2, lines 35-37), not on a task basis.

There is no motivation to combine Brown and Tuma because neither of them addresses the problem of task-based multiprocessor system. There is no teaching or suggestion that task entries having task status, task identifier, task start address, etc. or task cycle providing task ID, a task command, etc. is present. Brown, read as a whole, does not suggest the desirability of allocating cache memory or coordinating tasks in response to a task cycle.

In the present invention, the cited references do not expressly or implicitly suggest (1) a block allocation circuit to allocate blocks of cache memory used by at least one of the task, (2) a task coordinator to coordinate the tasks in response to a task cycle issued by the processor, (3) a task status, identifier, start address, block size, and task cache address; and (4) a task cycle providing the task ID, task command, start address, and block size. In addition, the Examiner failed to present a convincing line of reasoning as to why a combination of Brown and Tuma is an obvious application of a task-based multiprocessor system.

C. Claims 4-10, 14-20, and 24-30 Are Not Obvious over Brown in view of Tuma and further in view of Nagata

The Examiner rejected claims 4-10, 14-20 and 24-30 under 35 U.S.C. §103(a) as being unpatentable over Brown in view of Tuma and further in view of U.S. Patent No. 5,138,696 issued to Nagata ("Nagata").

Brown discloses a programmable data flow processor for performing data transfers as discussed above. Tuma discloses an address generator for solid state disk drive as discussed above.

Nagata discloses a printer provided with font memory card. A static random access memory (SRAM) includes a total font number storage area, a font attribute address table area, a free area, a character attribute address table, a bit map font area, and a free block retrieve table area (Nagata, col. 4, lines 10-42). The free block retrieve table area stores free block retrieve flags which represent information about the unused blocks (Nagata, col. 4, lines 43-45).

Brown, Tuma and Nagata, taken alone or in any combination, does not disclose, suggest, or render obvious (1) a block allocation circuit to allocate blocks of cache memory used by at least one of the task, (2) a task coordinator to coordinate the tasks in response to a task cycle issued by the processor, (3) a search logic circuit to locate a free block, and (4) a block information generator to generate block information.

Nagata merely discloses flags to indicate unused blocks among the bit map font blocks (Nagata, col. 4, lines 42-45). The font blocks are part of the SRAM. They do not correspond to data blocks in use in the cache memory as recited in claim 4, 14, and 24. Furthermore, none of Brown, Tuma, and Nagata discloses a search logic circuit to locate a free block and a block information generator to generate block information including a block size, a block starting address, and a block ending address.

There is no motivation to combine Brown, Tuma and Nagata because none of them addresses the problem of task-based multiprocessor system. There is no teaching or suggestion that a block allocation circuit/a block information generator or a task coordinator, or a search logic circuit is present. Brown, read as a whole, does not suggest the desirability of allocating cache memory or coordinating tasks in response to a task cycle.

In the present invention, the cited references do not expressly or implicitly disclose or suggest the claimed invention. In addition, the Examiner failed to present a convincing

line of reasoning as to why a combination of Brown, Tuma and Nagata is an obvious application of a task-based multiprocessor system.

In the final Office Action, the Examiner cited Nagata in column 4, lines 10-45 and Figs 23-24 to support the contention that Nagata discloses a search logic circuit (Final Office Action, page 4, paragraph 4(v)). However, nowhere in the cited portions and Figures that such a search logic circuit or its equivalent is disclosed. As discussed above, Nagata merely discloses a free block retrieve table area that stores free block retrieve flags which represent information about the unused blocks (Nagata, col. 4, lines 43-45). Nagata does not disclose or suggest locating a free block by shifting through a list of busy flags as recited in claims 4, 14, and 24.

In the final Office Action, the Examiner cited Brown, Figs 23, 24, and 11 to support the contention that Brown discloses a block information generator (Final Office Action, page 4, paragraph 4(v)). However, none of these figures shows a block information generator. The table shown in Figure 11 merely shows a memory mapping of the DSP's address to the generated address. Figure 23 merely shows a flowchart for the EXEC routine task program and a task list. Figure 24 merely shows a flowchart for the EXEC routine task scheduler and the task status. In contrast, a block information generator generates block information of a free block available for a new task as recited in claims 4, 14, and 24.

In the final Office Action, the Examiner merely directed Applicant's attention to the above sections without addressing the issue of motivation to combine (Final Office Action, page 5, paragraph 4(vii)). The Examiner also failed to present a convincing line of reasoning as to why a combination of the cited references is an obvious application of a task-based multiprocessor system.

Therefore, Applicant believes that claims 2-3, 12-13, and 22-23 are not obvious over Brown in view of Tuma, and dependent claims 4-10, 14-20, and 24-30 are not obvious over Brown in view of Tuma and further in view of Nagata.

Accordingly, claims 1, 11, and 21 and their respective dependent claims are distinguishable over the cited prior art references.

VIII. CONCLUSION

Applicant respectfully requests that the Board enter a decision overturning the Examiner's rejection of all pending claims, and holding that the claims are neither anticipated nor rendered obvious by the prior art.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: October 29, 2004

THINH V. NGUYEN
Reg. No. 42,034

12400 Wilshire Blvd., 7th Floor
Los Angeles, CA 90025-1026
(714) 557-3800

IX. CLAIMS APPENDIX

The claims of the present application which are involved in this appeal are as follows:

1. (original) An apparatus comprising:

a task table coupled to a bus interface to store task entries corresponding to tasks executed by at least one processor;

a block allocation circuit coupled to the bus interface and a cache memory to allocate blocks of the cache memory used by at least one of the tasks; and

a task coordinator to coordinate the tasks in response to a task cycle issued by the at least one processor.

2. (original) The apparatus of claim 1 wherein each of the task entries comprises at least one of a task status, a task identifier (ID), a task start address, a task block size, and a task cache address.

3. (original) The apparatus of claim 2 wherein the task cycle provides at least one of the task ID, a task command, the task start address, and the task block size.

4. (original) The apparatus of claim 3 wherein the block allocation circuit comprises:

a search logic circuit to locate a free block by shifting through a list of busy flags corresponding to data blocks in use in the cache memory; and

a block information generator coupled to the block search logic to generate block information of a free block available for a new task, the block information including at least a block size, a block starting address, and a block ending address.

5. (original) The apparatus of claim 1 wherein the task coordinator comprises:

a task table updater to update the task table; and

an address generator to generate address information to the cache memory.

6. (original) The apparatus of claim 3 wherein the task status is one of an invalid status, a shared status, and an exclusive status.

7. (original) The apparatus of claim 6 wherein the task command is one of a read shared command, a read exclusive command, a write command, and a flush command.

8. (original) The apparatus of claim 7 wherein the task block size corresponds to number of cache lines in one of the blocks.

9. (original) The apparatus of claim 8 wherein the bus interface is one of a processor bus interface and a multiprocessor bus interface.

10. (original) The apparatus of claim 9 wherein the cache memory is one of an internal cache and an external cache with respect to the at least one processor.

11. (original) A method comprising:

storing task entries corresponding to tasks executed by at least one processor in a task table;

allocating blocks of the cache memory used by at least one of the tasks; and

coordinating the tasks in response to a task cycle issued by the at least one processor.

12. (original) The method of claim 11 wherein each of the task entries comprises at least one of a task status, a task identifier (ID), a task start address, a task block size, and a task cache address.

13. (original) The method of claim 12 wherein the task cycle provides at least one of the task ID, a task command, the task start address, and the task block size.

14. (original) The method of claim 13 wherein allocating comprises:

locating a free block by shifting through a list of busy flags corresponding to data blocks in use in the cache memory; and

generating block information of a free block available for a new task, the block information including at least a block size, a block starting address, and a block ending address.

15. (original) The method of claim 11 wherein coordinating the tasks comprises:

updating the task table; and

generating address information to the cache memory.

16. (original) The method of claim 13 wherein the task status is one of an invalid status, a shared status, and an exclusive status.

17. (original) The method of claim 16 wherein the task command is one of a read shared command, a read exclusive command, a write command, and a flush command.

18. (original) The method of claim 17 wherein the task block size corresponds to number of cache lines in one of the blocks.

19. (original) The method of claim 18 wherein the bus interface is one of a processor bus interface and a multiprocessor bus interface.

20. (original) The method of claim 19 wherein the cache memory is one of an internal cache and an external cache with respect to the at least one processor.

21. (original) A system comprising:

a processor bus; and

a plurality of processors coupled to the processor bus, each of the plurality of processors having an internal task manager and an internal cache memory, the internal task manager comprising:

a task table coupled to a bus interface to store task entries corresponding to tasks executed by at least one processor,

a block allocation circuit coupled to an internal bus interface and the interval cache memory to allocate blocks of the interval cache memory used by at least one of the tasks, and

a task coordinator to coordinate the tasks in response to a task cycle issued by the at least one processor.

22. (original) The system of claim 21 wherein each of the task entries comprises at least one of a task status, a task identifier (ID), a task start address, a task block size, and a task cache address.

23. (original) The system of claim 22 wherein the task cycle provides at least one of the task ID, a task command, the task start address, and the task block size.

24. (original) The system of claim 23 wherein the block allocation circuit comprises:

a search logic circuit to locate a free block by shifting through a list of busy flags corresponding to data blocks in use in the internal cache memory; and

a block information generator coupled to the block search logic to generate block information of a free block available for a new task, the block information including at least a block size, a block starting address, and a block ending address.

25. (original) The system of claim 21 wherein the task manager comprises:

a task table updater to update the task table; and

an address generator to generate address information to the interval cache memory.

26. (original) The system of claim 23 wherein the task status is one of an invalid status, a shared status, and an exclusive status.

27. (original) The system of claim 26 wherein the task command is one of a read shared command, a read exclusive command, a write command, and a flush command.

28. (original) The system of claim 27 wherein the task block size corresponds to number of cache lines in one of the blocks.

29. (original) The system of claim 28 wherein the processor bus is a multiprocessor bus.

30. (previously presented) The system of claim 29 further comprising:

a external cache memory; and

an external task manager coupled to the external cache memory and the processor bus, the external task manager comprising:

an external task table to store task entries corresponding to tasks executed by at least one processor,

a external block allocation circuit coupled to the processor bus and the external cache memory to allocate blocks of the external cache memory used by the tasks, and

an external task coordinator to coordinate the tasks in response to the task cycle issued by the at least one processor.